


Article

# A Lightweight and Partitioned CNN Algorithm for Multi-Landslide Detection in Remote Sensing Images

Peijun Mo, Dongfen Li \*, Mingzhe Liu \* , Jiaru Jia and Xin Chen

State Key Laboratory of Geohazard Prevention and Geoenvironment Protection,  
Chengdu University of Technology, Chengdu 610059, China; mopeijun@stu.cdut.edu.cn (P.M.);  
2021020863@stu.cdut.edu.cn (J.J.); 2021020862@stu.cdut.edu.cn (X.C.)

\* Correspondence: lidongfen17@cdut.edu.cn (D.L.); liumz@cdut.edu.cn (M.L.)

**Abstract:** Landslide detection is crucial for natural disaster risk management. Deep-learning-based object-detection algorithms have been shown to be effective in landslide studies. However, advanced algorithms currently used for landslide detection require high computational complexity and memory requirements, limiting their practical applicability. In this study, we developed a high-resolution dataset for landslide-prone regions in China by extracting historical landslide remote sensing images from the Google Earth platform. We propose a lightweight LP-YOLO algorithm based on YOLOv5, with a more-lightweight backbone that incorporates our designed PartitionNet and neck equipped with CSPCrossStage. We constructed and added the vertical and horizontal (VH) block to the backbone, which explores and aggregates long-range information with two directions, while consuming a small amount of computational cost. A new feature fusion structure is proposed to boost information flow and enhance the location accuracy. To speed up the model learning process and improve the accuracy, the SCYLLA-IoU (SIoU) bounding box regression loss function was used to replace the complete IoU (CIoU) loss function. The experimental results demonstrated that our proposed model achieved the highest detection performance (53.7% of Precision, 49% of AP50 and 25.5% of AP50:95) with a speed of 74 fps. Compared to the YOLOv5 model, the proposed model achieved 4% improvement for Precision, 2.6% improvement for AP50, and 2.5% for AP50:95, while reducing the model parameters and FLOPs by 38.4% and 53.1%, respectively. The results indicated that the proposed lightweight method provides a technical guidance for achieving reliable and real-time automatic landslide detection and can be used for disaster prevention and mitigation.

**Keywords:** landslide detection; remote sensing image; deep learning; LP-YOLO; YOLOv5



**Citation:** Mo, P.; Li, D.; Liu, M.; Jia, J.; Chen, X.; A Lightweight and Partitioned CNN Algorithm for Multi-Landslide Detection in Remote Sensing Images. *Appl. Sci.* **2023**, *13*, 8583. <https://doi.org/10.3390/app13158583>

Academic Editor: Jianbo Gao

Received: 28 April 2023

Revised: 18 July 2023

Accepted: 20 July 2023

Published: 25 July 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Landslides are severe geological hazards that widely occur in mountainous environments with slopes and frequently lead to chain reactions such as mountain collapses and debris flows, which can pose serious risks to human life and property. Therefore, enhancing the detection and early warning systems for landslide-related geological catastrophes holds considerable implications in the context of China's endeavors towards disaster mitigation and risk reduction [1,2].

Traditional landslide detection methods primarily rely on geologists, which often entails significant manpower and financial investments. However, the effectiveness of these methods may not always meet expectations. In light of the advancements in satellite imaging accuracy, researchers have increasingly proposed landslide detection approaches that leverage optical image data. Concurrently, machine learning has become increasingly popular in the field of landslide detection. Besides, the emergence of convolutional neural networks (CNNs) has led to the successful application of deep-learning-based object recognition algorithms in landslide detection, and they have gradually become mainstream. In contrast to machine learning approaches, deep learning techniques abandon the complicated artificially designed features, which adopt deeper convolutional neural networks to

automatically acquire distinguishing characteristics. Furthermore, the data sample capacity of deep learning for landslide detection can be extensive, rendering it more appropriate for large-scale landslide identification and endowing it with a more-robust generalization capability. Although deep-learning-based algorithms have shown success in detecting landslides, they still face some challenges: These models necessitate substantial computational resources and numerous parameters, leading to diminished inference efficiency. As a result, employing them in power-limited contexts or embedded platforms with the objective of achieving real-time detection becomes challenging. Moreover, the scarcity of open-source repositories containing high-spatial-resolution images of landslides hampers the effective training and validation of these models.

To tackle these challenges, we constructed a landslide dataset utilizing the Google Earth platform in this study. This paper presents a lightweight framework named LP-YOLO, which achieves real-time landslide detection. Our contributions are as follows:

(1) We propose the Partition module and form a new feature extraction network named PartitionNet to replace the backbone of YOLOv5, which brings better performance, while reducing drastically the redundant parameters and computational complexity.

(2) A new feature-exploiting module named the VH block is constructed and added to the backbone to retain the information after down-sampling and to explore long-range information with a small computational cost.

(3) We designed a new feature fusion structure and propose a CSPCrossStage module instead of C3 in the neck of the model to boost information flow and enhance the location accuracy of multi-scale landslides with less computing resource.

(4) The SIOU loss function and the attention mechanisms were introduced to expedite the convergence speed during training and enhance the detection accuracy of the model.

## 2. Related Work

Landslide detection can generally be categorized into two approaches: traditional methods of landslide identification and automatic identification methods based on machine learning algorithms.

Traditional methods of landslide detection often rely on field surveys conducted by experienced geologists, complemented by instrumental imaging techniques for analysis. These methods involve on-site inspections, geological mapping, and the collection of ground truth data, for example using interferometry synthetic aperture radar (InSAR) technology to obtain multi-temporal data to observe whether the slope is deformed, which can be used as a basis to infer potential landslides [3]. While traditional methods have been widely practiced and have proven effective, they have the limitations of being time-consuming and resource-intensive.

The second category predominantly utilizes pre-existing datasets of landslides and facilitates automatic identification through the construction of algorithmic models. Generally, automatic landslide detection techniques can be categorized into machine learning approaches and deep learning approaches. Machine learning algorithms encompass methods such as Bayesian, logistic regression, support vector machines (SVMs), random forests, and artificial neural networks [4–7], which can utilize various features related to landslide occurrence, such as texture and terrain information for classification and prediction. For instance, Pourghasemi [8] applied random forest to evaluate the sensitivity of landslides, and Tien [9] utilized SVM and kernel logistic regression for landslide recognition. Artificial neural networks, including pulse-coupled neural networks (PCNNs) and spiking neural networks, have been shown to possess outstanding capabilities in image fusion and computer vision applications [10,11].

Owing to the swift advancements in hardware equipment and artificial intelligence, deep learning techniques have emerged as an additional potent data-driven approach for detection. Consequently, a multitude of sophisticated object-detection algorithms have surfaced, including two-stage object-detection algorithms represented by region-based convolutional neural networks (RCNNs), Fast R-CNN, and Faster R-CNN [12,13] and

single-stage object detection networks represented by the SSD algorithm [14] and the YOLO algorithm series [15–18]. For instance, Ju [19] selected the YOLOv3 and Mask RCNN algorithms to achieve automatic recognition of loess landslides, with an optimal average precision of 18.9%. Ji [20] proposed an enhanced convolutional neural network for landslide detection of Bijie City, which demonstrated the effectiveness of a new landslide prediction based on the dataset. Hou [21] incorporated a coordinate attention mechanism [22] to enhance the YOLOX [23] object-detection model, effectively tackling the problem of the poor detection of complex mixed landslides. Tang [24] proposed SegFormer, a model based on the Transformer architecture, which is capable of automatically detecting landslides.

In recent years, various lightweight neural architectures such as MobileNetV1-3 [25–27], ShuffleNet [28], GhostNet [29], and FasterNet [30] have followed, aiming to achieve fewer parameters, a fast inference speed, and high performance. In brief, MobileNet incorporates depthwise separable convolution and an inverted residual structure to decrease the computational expense while simultaneously improving the detection performance. ShuffleNet substitutes the  $1 \times 1$  convolution with group convolution and incorporates a shuffle operation to facilitate information flow among various groups. In GhostNet, to avoid redundant features maps, the spatial features are only captured by inexpensive operations for half of the features. FasterNet proposes a partial convolution to extract features with an efficient and parameter-friendly manner. Besides, residual connections [31] and dense connections [32] are widely used in these network designs to alleviate gradient degradation problems and aggregate features with diverse receptive fields. Inspired by these works, our method LP-YOLO proposes a novel Partition module and Partition block to construct a lightweight feature-extraction backbone by combining the residual and dense connections, which is used to diminish the computational overhead and fulfill the demands of real-time detection tasks.

### 3. Method

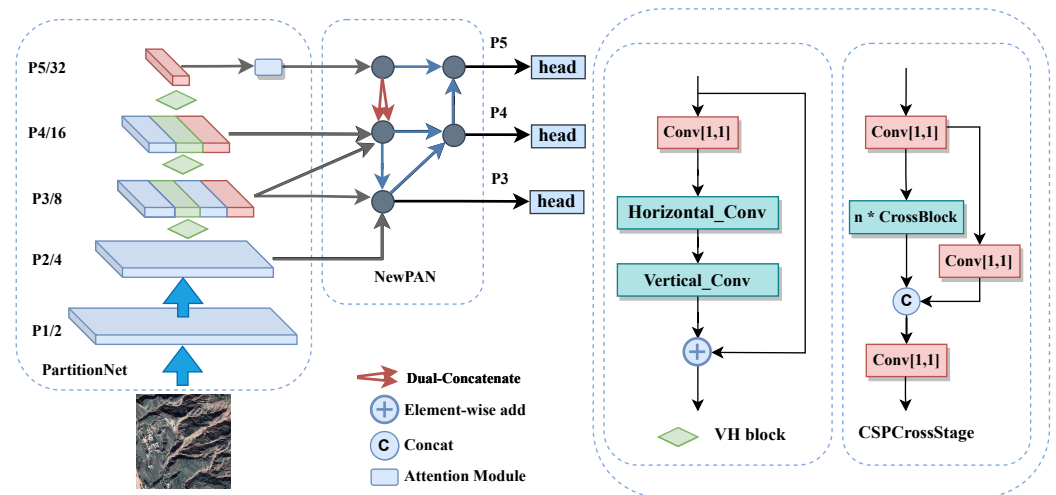
In this section, we begin by reviewing our baseline model YOLOv5 and then provide a comprehensive description of the LP-YOLO model (Figure 1). We discuss the network structure, the VH block, the new PAN feature fusion structure, the loss function, and the attention module.

#### 3.1. A Brief Review of YOLOv5

YOLOv5 is composed of three primary elements: the CSPDarkNet backbone with the C3 block, the path aggregation network (PAN) [33] neck with the spatial pyramid pooling feature (SPPF) layer [34], and the detection head. The C3 block contains three general convolutional and bottleneck modules and was inspired by CSPNet [35]. Within this block, the feature map is bifurcated into two sections: one segment traverses the bottleneck module, while the other is conveyed to the convolutional module and subsequently merged with the first portion. The PAN structure facilitates information flow and feature aggregation from bottom to top, and the SPPF layer in YOLOv5 serves to aggregate multi-scale contextual information from feature maps. Various data augmentation methods, such as mosaic augmentation, are employed within the model to mitigate data imbalance issues associated with small, medium, and large objects present in the dataset. In addition, the model employs the complete intersection over union (CIoU) [36] bounding box regression loss function for optimizing the model's ability to accurately localize objects.

#### 3.2. Lightweight LP-YOLO Model

As shown in Figure 1, the following characteristics can be identified from our proposed LP-YOLO model.



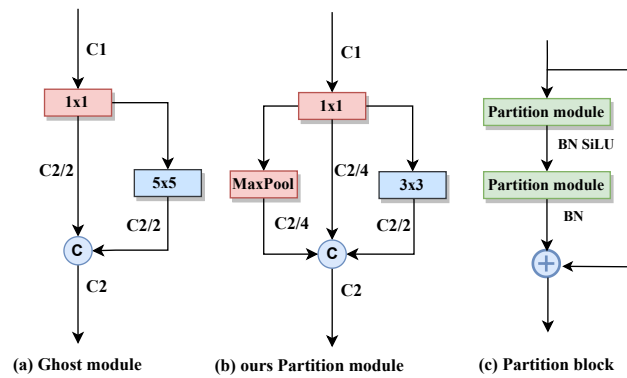
**Figure 1.** Framework of our proposed LP-YOLO. The backbone is PartitionNet; the neck is the new path aggregation network. The VH block is proposed to enhance information flow after the down-sampling layer.

### 3.2.1. Backbone

Given that the original YOLOv5 model utilizes CSPDarkNet as its backbone for feature extraction, which comprises numerous regular convolutional layers and demands significant computational resources, we propose a lightweight network called PartitionNet to supplant CSPDarkNet.

The Partition module is shown in Figure 2b. Firstly, we make a comparison with the Ghost module and simplify the Ghost module in Figure 2a, where C1 and C2 represent the quantity of the input channels and output channels, respectively. The Ghost module aims to reduce the number of  $1 \times 1$  convolutional kernels by half during the convolutional operations, followed by an inexpensive operation that generates a set of transformed features. The remaining half of the feature maps and transformed features are concatenated to generate output features, thereby reducing the number of nonlinear operations and significantly diminishing the computational resources and parameters. However, the intrinsic features are just produced by half of pointwise convolution, and the spatial information is only captured by inexpensive operations with depthwise convolution. This method may lead to weak spatial information capturing, which could hinder detection performance.

To address this challenge, we propose a module in a partitioned way, named the Partition module. It consists of three components: pointwise convolution, max-pooling layer, and depthwise convolution. We set a kernel size of 3 and a stride of 1 in the max-pooling layer, designed to eliminate redundant information and enhance the robustness of extracted features. In order to avoid feature compression and make the subsequent concatenation convenient, we set a padding of 1. Besides, we leveraged the  $3 \times 3$  depthwise convolution and  $1 \times 1$  convolution to decrease the parameters in our network. More importantly, we separated the C2 into four parts, with a C2 ratio of  $1 \times 1$  convolution, a max-pooling layer, and  $3 \times 3$  depthwise convolution being 1, 1, and 2, respectively. This allowed us to use only a quarter of the convolutional kernels to produce features instead of half of the Ghost module and adding the max-pooling layer to alleviate the information loss caused by direct splicing with the original feature map. Through this approach, we alleviated the computational burden and achieved improved performance, as demonstrated by our experiments.



**Figure 2.** The Partition module and Partition block.

Inspired by MobileNetV2, our Partition block is also an inverted bottleneck. It is shown in Figure 2c and composed of two Partition modules stacked in series. The first Partition module functions as an amplification layer to augment the number of channels, while the second one squeezes the output channels to align with the residual connection. We incorporated a down-sampling function utilizing a  $3 \times 3$  depthwise convolution with a stride of 2. In addition, we utilized the sigmoid-weighted linear unit (SiLU) [37] activate function to introduce nonlinearity into the network to enhance the network’s generalization ability. Compared to ReLU, SiLU has self-stability and absorbs the advantages of ReLU, which is smoother around zero so that negative values also have slight activation, and at the same time, it plays a balancing role on the gradient with a larger value. The formula of SiLU is defined as:

$$SiLU(x) = x \cdot sig(x) \tag{1}$$

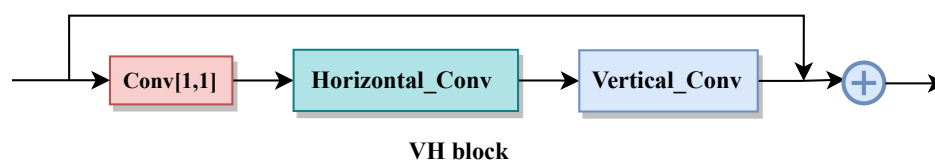
where  $sig$  and  $x$  are represented as the sigmoid function and input value, respectively. Finally, the backbone of LP-YOLO is constructed by stacking the Partition block and VH block, named PartitionNet. The specific details are provided in Table 1, which contains six columns.

**Table 1.** The architecture of the backbone of LP-YOLO. P-Block denotes the Partition block; Exp Size and Out Size refer to the size of the expansion channels and output channels, respectively. SE [38] denotes whether or not an SE block is employed in P-Block.

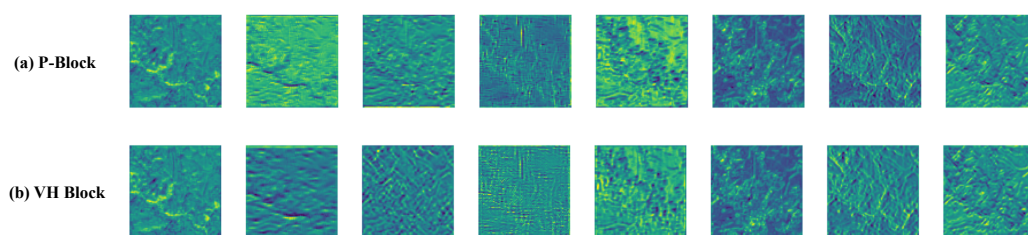
Input	Operator	Exp Size	Out Size	SE	Stride
$640^2 \times 3$	Conv2d $3 \times 3$	-	8	-	2
$320^2 \times 8$	P-Block	16	8	-	1
$320^2 \times 8$	P-Block	48	16	-	2
$160^2 \times 16$	P-Block	72	16	-	1
$160^2 \times 16$	VH block	-	16	-	-
$160^2 \times 16$	P-Block	72	24	1	2
$80^2 \times 24$	P-Block	120	24	1	1
$80^2 \times 24$	VH block	-	24	-	-
$80^2 \times 24$	P-Block	240	40	-	2
$40^2 \times 40$	P-Block	184	40	-	1
$40^2 \times 40$	P-Block	480	56	1	1
$40^2 \times 56$	P-Block	480	56	1	1
$40^2 \times 56$	VH block	-	56	-	-
$40^2 \times 56$	P-Block	672	80	-	2
$20^2 \times 80$	P-Block	960	80	-	1
$20^2 \times 80$	P-Block	960	80	1	1
$20^2 \times 80$	P-Block	960	80	-	1
$20^2 \times 80$	P-Block	960	80	1	1

### 3.2.2. VH Block for Feature Enhancement

In PartitionNet, the down-sampling layer is achieved by depthwise convolution. Depthwise convolution is a type of group convolution where the convolutional kernel is responsible for capturing information only from specific channels in the input feature map. This result in no information interaction between groups and a loss of feature information to some extent, especially at the edges. To address this limitation, we propose a VH block (shown in Figure 3), which was inspired by cross-convolution [39]. As shown in Figure 4, the VH block explores long-range information for feature enhancement in the horizontal and vertical axes, compensating for the lack of information exchange between groups and consuming few computational resources. By integrating the VH module into every down-sampling layer of the backbone, we can improve landslide detection performance, as landslide recognition heavily relies on texture and edge features.



**Figure 3.** The framework of VH block.



**Figure 4.** The feature visualization of VH block.

### 3.2.3. Using New PAN Feature Fusion Structure and CSPCrossStage

The PAN (Figure 5a) serves as a neck to extract multi-scale feature maps in the YOLOv5 model. The homogeneity of landslide areas and their surroundings, often characterized by similar color, brightness, and texture, presents challenges for accurate landslide detection. However, for landslide detection, texture and edge features extracted by shallow convolutions are more important than other objects. To alleviate potential losses of landslide edge information, we enhanced the existing PAN feature structure by incorporating the feature maps from after the second subsampling layer P2 and reducing the dimensionality of the P3 layer for convenient feature concatenation in the next stage. Additionally, the dual-concatenate operation was used in the layer of P5 for feature enhancement, and the dual-concatenate operation involves concatenating the up-sampling layer of P5 twice in the feature fusion layer. We found that it demonstrated significant effectiveness in detecting medium- and small-sized landslides. In comparison to the previous PAN, the enhanced feature fusion structure, NewPAN (Figure 5b), improved the location accuracy of multi-scale landslides and achieved a 0.1% Precision improvement, a 0.8% Recall improvement, a 0.6% AP50 improvement and a 0.4% AP50:95 improvement, with a detection performance of 49% AP50 and 25.5% AP50:95.

In addition, as shown in Figure 6, we replaced the C3 module with CSPCrossStage in the neck to further reduce the computation and parameters. The structure is similar to C3, the channel split and cross-stage dense connections were used for CSPCrossStage to improve the network's feature representation capability.

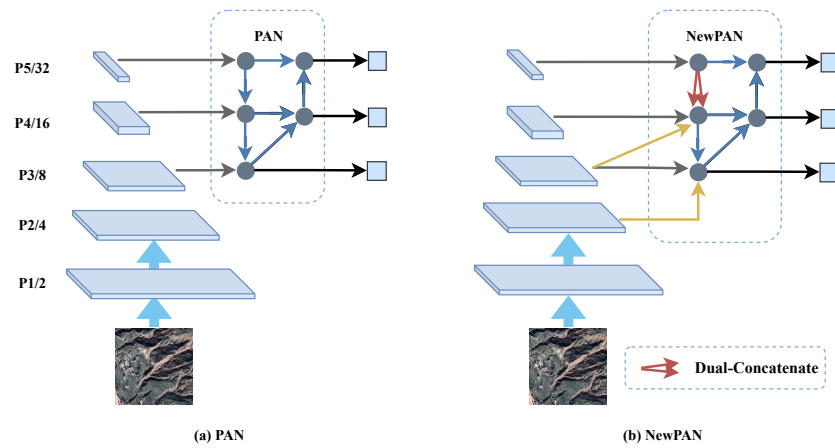


Figure 5. (a) PAN structure. (b) NewPAN structure.

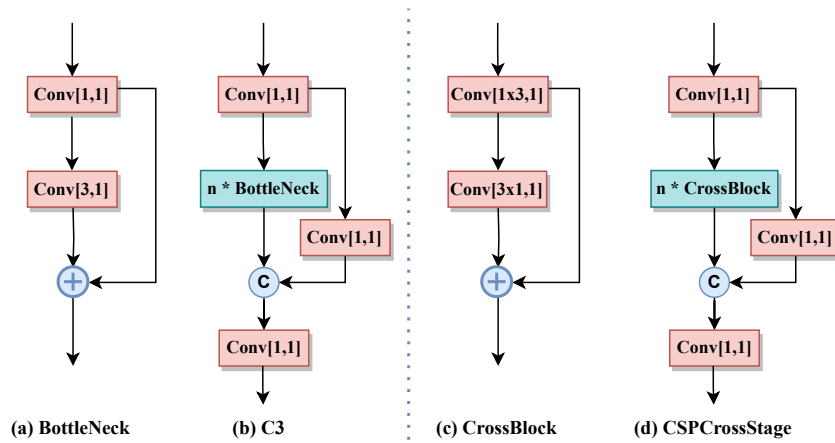


Figure 6. The framework of BottleNeck, C3, and proposed cross-block, CSPCrossStage.

The compression ratio of the parameters (ignoring the weights of the BN layer) of CrossBlock and BottleNeck can be calculated as:

$$\begin{aligned}
 r &= \frac{CrossBlock_{Params}}{BottleNeck_{Params}} \\
 &= \frac{2 \cdot C_{in} \cdot C_{out} \cdot k \cdot 1}{C_{in} \cdot C_{out} \cdot 1 \cdot 1 + C_{in} \cdot C_{out} \cdot k \cdot k} \\
 &= \frac{2k}{k^2 + 1}
 \end{aligned}
 \tag{2}$$

Table 2 compares the parameter counts of C3 and CSPCrossBlock under similar conditions. We set  $k$  (kernel size) to 3 in our model, and then, the compression ratio  $r = 0.6$  means that CrossBlock has 40% fewer parameters than C3.

Table 2. Comparison of the parameters of C3 and CSPCrossBlock. Input/Output Size means the number of input/output channels. Numbers means the number of modules.

Module	Numbers	Input Size/Out Size	Parameters (M)
C3	1	256/256	0.296
CSPCrossBlock	1	256/256	0.230
BottleNeck	1	256/256	0.656
CrossBlock	1	256/256	0.394

### 3.2.4. Using SIoU Loss Function and Attention Mechanism

To further improve the accuracy, the box regression loss function is another aspect to be considered. The intersection over union (IoU) is a metric utilized in object detection tasks to quantify the similarity between two bounding boxes. The IoU loss function (IoU loss), derived from the IoU metric, is employed as the regression loss function for bounding boxes in object-detection models. The IoU metric serves two primary functions: distinguishing positive and negative samples and assessing the correlation between the predicted box and ground truth box (GT box). For better bounding box regression of the model, a series of metrics (GIoU [40], DIoU, CIoU) have been developed. The YOLOv5 model incorporates the CIoU loss as the bounding box regression loss function to assess the predicted results. The CIoU metric considers factors such as the distance, overlap area, and aspect ratio between the predicted and the GT boxes. However, it does not account for the direction of the mismatch between the predicted and GT box, which can result in slower convergence during the training process [41]. Therefore, to make up for these deficiencies, we introduced the SIoU loss in LP-YOLO. In detail, the SIoU loss consists of 4 cost contributions: angle cost, distance cost, shape cost, and IoU cost. The formula of the SIoU loss is defined as:

$$L_S = 1 - \text{IoU} + \frac{\Delta + \Omega}{2} \tag{3}$$

$$= 1 - \text{IoU} + \frac{\sum_{t=x,y}(1 - e^{-\gamma\rho_t}) + \sum_{t=w,h}(1 - e^{-\omega_t})^\theta}{2}$$

where  $\Omega = \sum_{t=w,h}(1 - e^{-\omega_t})^\theta$ ; this denotes the shape cost, which is used to consider the differences in the aspect ratio between the predicted and GT box.  $\omega_t$  means the ratio of the difference between the width of the GT box and the predicted box to the maximum width. The value of  $\theta$  regulates the degree of emphasis placed on the shape loss. In the LP-YOLO model, we set  $\theta$  equal to 4.  $\Delta = \sum_{t=x,y}(1 - e^{-\gamma\rho_t})$ ,  $\gamma = 2 - \varphi$ ; this denotes the distance cost and is based on penalty metrics of the angle cost. The metric of the angle cost can be defined as:

$$\varphi = 1 - 2 \cdot \sin^2(\arcsin(x) - \frac{\pi}{4}) \tag{4}$$

where  $x = \frac{c_h}{\sigma} = \sin(\alpha)$ , as shown in Figure 7,  $B$  and  $B^{gt}$  represent the centroids of the predicted and the GT box, respectively, where  $\sigma$  denotes the Euclidean distance between the points, and  $\alpha$  represents the angle between  $\alpha$  and the X axis.  $c_h$  denotes the difference in height between the centroids of the two bounding boxes. When the predicted box does not intersect with the GT box, the model endeavors to align the prediction with the nearest axis (if  $\alpha < \beta$ , minimize  $\alpha$ ; otherwise, minimize  $\beta$ ) before adjusting it to prevent model confusion.

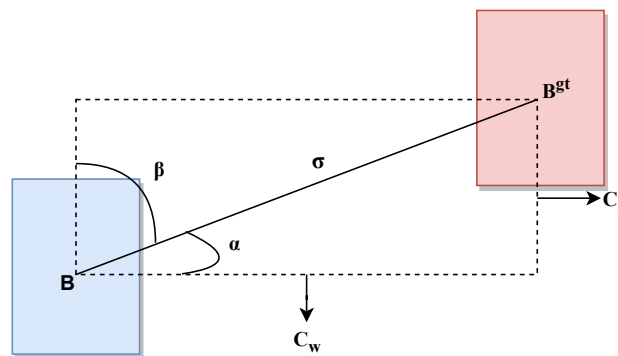


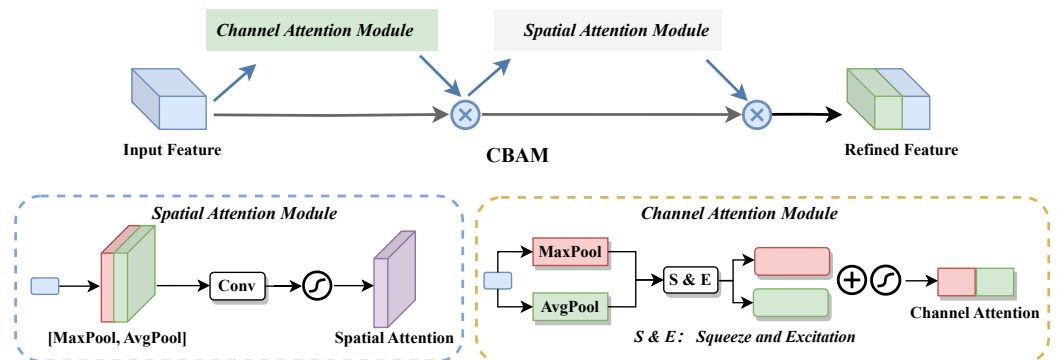
Figure 7. Angle cost contribution.

The experiments demonstrated that the overall error of the SIoU loss function was more heavily influenced by the number of iterations and ultimately attained a lower value.



Applying the SIoU loss to the landslide dataset can achieve a faster convergence speed during the training process and improve the detection performance.

In order to increase the model's capacity to prioritize crucial landslide information, suppress redundant background information, and improve detection performance, we incorporated the lightweight CBAM module [42] into the backbone before the SPPF layer. As illustrated in Figure 8, the CBAM module integrates both the spatial attention and channel attention mechanisms to enable the model to selectively attend to pixel regions in the image that are critical for accurate classification. By incorporating CBAM into the backbone, the model is better able to focus on landslide regions and enhance detection accuracy.

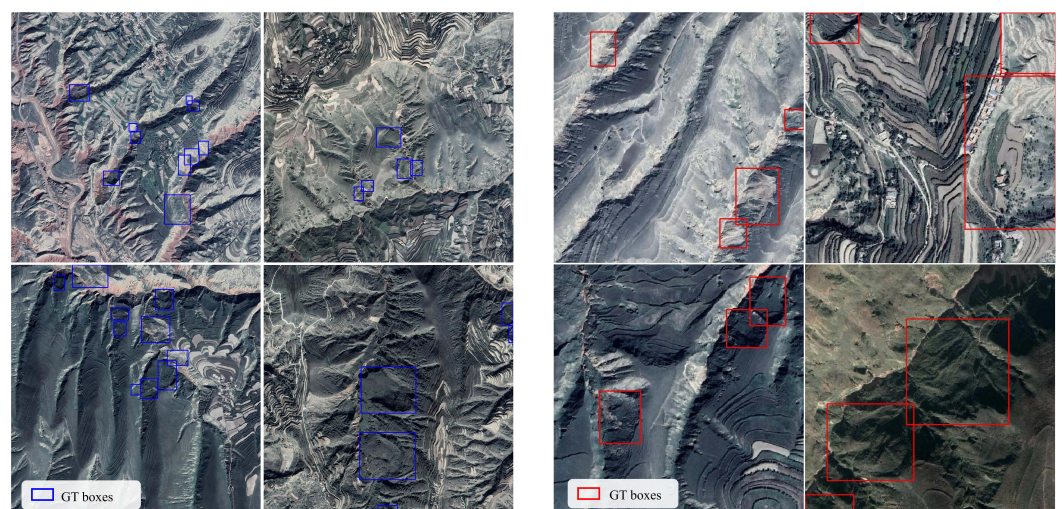


**Figure 8.** The framework of CBAM.

## 4. Experiments

### 4.1. Datasets' Collection

The study area is situated at the junction of southern Gansu Province and eastern Qinghai Province, characterized by a dry climate, low average annual precipitation, and sparse vegetation. To identify landslides in this region, we obtained remote sensing images from the Google Earth Engine platform. As the original images were too large and did not meet the requirements for object detection, we used a Python library to crop them into JPEG format images with a size of  $2000 \times 2000$  px. As shown in Figure 9, each image was then annotated and visually explained for landslide identification. In this study, a total of 15,301 landslides were marked.



**Figure 9.** Original datasets. GT boxes denote the real landslide area.

The cross-cutting method refers to dividing an image into several overlapping patches with a certain stride, and each patch can be regarded as a small image for further processing. In this study, the cross-cutting method was used to divide the  $2000 \times 2000$  px images into

several  $640 \times 640$  px images to maintain the integrity of the image texture. This methodology effectively enhances the utilization of image information and avoids information loss caused by image compression. After processing, the dataset was partitioned into training and validation sets at an 8:2 ratio, with 5434 for the training set and 1461 for the validation set obtained after screening. It comprised two categories: landslides and non-landslides. Figure 9 showcases the processed remote sensing images with resolutions of  $2000 \times 2000$  px (left) and  $640 \times 640$  px obtained by cropping (right).

#### 4.2. Evaluation Metrics

To evaluate the detection performance of the model on landslide images, a series of evaluation metrics was utilized in the experiments. These metrics included the Precision ( $P$ ), Recall ( $R$ ), AP50, AP50:95, floating point operations (FLOPs), Latency, and the number of parameters (Params).

Precision ( $P$ ) and Recall ( $R$ ) are essential decision scores used to assess the quality of detection models. The scores are typically defined in terms of three metrics. These are defined as:

$$P = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN} \quad (5)$$

where  $TP$  means true positive (number of correctly predicted landslide samples),  $FP$  means false positive (the number of predicted non-landslide objects regarded as landslides), and  $FN$  denotes false negatives (the number of undetected landslides).  $P$  indicates the ratio of true positive predictions to the total number of landslide samples predicted as positive, which can evaluate the false detection rate.  $R$  denotes the ratio of true positive predictions to all actual positive samples in the dataset, which is used to detect how sensitively the model identified the landslide. The average precision ( $AP$ ) can be obtained by the  $PR$  curve, and it is defined as:

$$AP = \int_0^1 P(R)d(R) \quad (6)$$

AP50 represents the  $AP$  calculated at an intersection over union (IoU) threshold of 50%. On the other hand, AP50:95 refers to the calculation of the  $AP$  under threshold conditions ranging from 50% to 95%, and it is defined as:

$$AP_{50:95} = \frac{1}{10} \sum_n AP_n \quad (7)$$

The number of parameters (Params) is an essential indicator of the model's space complexity, which influences the amount of memory used. FLOPs refers to the number of floating point operations performed by the model, which is used as a metric of computational complexity. Latency represents the inference time on a single image for the model.

#### 4.3. Training Strategy

In the experiment, an input image size of  $640 \times 640$  was used, with an initial learning rate of 0.01 and a batch size of 16, and the Adam optimizer was selected. The Mosaic data augmentation technique was employed for data pre-processing, which involves concatenating four images and randomly applying scaling, cropping, and shuffling the images. The number of training epochs was set to 150, as it was found that stability could be achieved after around 130 epochs based on the experimental results from the landslide dataset. We performed all the experiments using the PyTorch 1.8 deep learning framework on a Linux system with hardware equipped with an NVIDIA Tesla P100-16GB GPU. The training process of LP-YOLO is shown in Algorithm 1.

**Algorithm 1:** Pseudocode of LP-YOLO algorithm.

---

**Input:** landslide remote sensing images, labels for bounding box coordinates  
object class,  $B_x$ ,  $B_y$ , width  $B_w$ , height  $B_h$

**Output:** Class probabilities  $P_c$  and predicted bounding box coordinates

- 1 Initialize landslide dataset = train images (80%) + validation images (20%);
- 2 batch size = 16;
- 3 epochs  $E = 150$ ;
- 4 data augmentation = true;
- 5 **Training stage:**
- 6 **foreach** *epoch* in  $E$  **do**
- 7     Load YOLO hyperparameters;
- 8     Randomly select *batch size* images;
- 9     Training on the landslide images with LP-YOLO algorithm (Figure 1);
- 10    Calculate the loss and update the parameters by backpropagation;
- 11    Evaluating detection performance of the algorithm using validation images;
- 12 **end**
- 13 save optimal bestweight.pt
- 14 **Testing stage:**
- 15 Use the trained LP-YOLO model to predict the test images;
- 16 predict  $P_c$ ,  $B_x$ ,  $B_y$ ,  $B_w$ , and  $B_h$ ;
- 17 display class  $P_c$  and predicted bounding box

---

## 5. Results

### 5.1. Ablation Experiments' Results

The detection performance changes caused by modifications in the network structure were evaluated through ablation experiments, and the results are presented in Table 3.

**Table 3.** Ablation study of LP-YOLO on landslide dataset val.

Model	AP50(%)	AP50:95(%)	Params (M)	FLOPs (G)
YOLOv5s baseline model	46.4	23.0	7.02	15.9
YOLOv5s baseline model + SIoU loss	47.9 (+1.5)	24.4 (+1.4)	7.02	15.9
LP-YOLO + PartitionNet (backbone)	46.8 (+0.4)	24.6 (+1.6)	4.51	6.9
+VH block and attention mechanism	48.4 (+1.6)	25.1 (+0.5)	4.53	7.2
+NewPAN (+CSPCrossStage)	49.0 (+0.6)	25.5 (+0.4)	4.34	7.4

In contrast to the YOLOv5s model, our experiments showed that using the SIoU loss function could achieve better location results and accuracy, for which the Precision increased from 49.7% to 51.9%, AP50 increased from 46.4% to 47.9%, and AP50:95 increased from 23.0% to 24.4%. Additionally, using only our proposed network PartitionNet to replace the backbone of the model led to a 38.2% reduction in the quantity of parameters and a 53.4% reduction in the FLOPs, while increasing the Precision, AP50, and AP50:95. With the introduction of the CBAM attention module and our proposed VH block, we obtained a 1.6% AP50 improvement and a 0.5% AP50:95 improvement with slightly increased Params and FLOPs compared to the model of the backbone of PartitionNet. Furthermore, using the NewPAN feature fusion structure and CSPCrossStage resulted in a 38.1% reduction in

the quantity of parameters compared to the YOLOv5s baseline model and a 4.2% reduction compared to the proposed model that used the previous PAN structure.

In addition, using NewPAN in the proposed model could achieve a 0.6% improvement for AP50 and a 0.4% improvement for AP50:95.

With the simultaneous improvement of the backbone, neck, and feature fusion structure and the introduction of the attention module, the proposed model achieved a reduction of 38.1% and 53.4% in the number of parameters and FLOPs, respectively, when compared to the baseline model. Furthermore, the model's accuracy was significantly improved, achieving an AP50 and AP50:95 of 49.0% and 25.5%, respectively.

### 5.2. Comparison of Experiments' Results with Different Backbones

To provide a more quantitative validation of the detection accuracy of our proposed method, we conducted a set of experiments comparing it to existing mainstream lightweight object-detection algorithms. We tested several lightweight backbone architectures, including ShuffleNet, PP-LCNet [43], EfficientNetB0 [44], MobileNetV3(Small), repVGG [45], GhostNet, and our PartitionNet with the same neck structures and loss functions.

As shown in Table 4, while the models with ShuffleNet, MobileNetV3, and PP-LCNet as backbones exhibited favorable performance with respect to the number of parameters, FLOPs, and Latency metrics on the landslide dataset, our algorithm outperformed them in terms of the P, R, AP50, and AP50:95 accuracy metrics.

Furthermore, compared to the CIoU loss function, we found that employing the SIoU loss function in YOLOv5s resulted in a 1.5%, 1.4%, and 2.2% improvement for AP50, AP50:95, and Precision, respectively. When comparing YOLOv5s-SIoU-PAN with LP-YOLO-PAN, we observed a decrease in AP50, but not exceeded 1.1%, and an increase in Latency, but a decrease in the parameters and FLOPs of 35.6% and 56.3%, respectively. Finally, we found that incorporating a VH block and attention module to the backbone, as well as replacing the PAN module with NewPAN resulted in an increase in the AP and a decrease in the Params when compared to LP-YOLO-PAN.

**Table 4.** Detection performance comparison of different backbone networks. LF denotes loss function.

Model	Backbone	Neck	LF	AP50 (%)	AP50:95 (%)	P (%)	R (%)	Latency (ms)	FLOPs (G)	Params (M)
YOLOv5-s	CSPDarkNet	PAN	CIoU	46.4	23.0	49.7	49.8	8.7	15.8	7.02
YOLOv5-s	CSPDarkNet	PAN	SIoU	47.9	24.4	51.9	48.6	8.8	15.8	7.02
YOLOv5-s	ShuffleNet	PAN	SIoU	44.3	21.9	49.3	46.7	8.2	1.8	0.84
YOLOv5-s	PP-LCNet	PAN	SIoU	43.9	20.8	50.3	46.1	9.4	5.8	2.96
YOLOv5-s	EfficientNetB0	PAN	SIoU	45.7	22.8	50.2	48.7	10.5	7.1	3.51
YOLOv5-s	MobileNetV3	PAN	SIoU	42.0	20.5	48.8	45.0	10.8	2.5	1.38
YOLOv5-s	repVGG	PAN	SIoU	44.5	21.5	50.1	47.2	9.1	7.0	3.4
YOLOv5-s	GhostNet	PAN	SIoU	45.6	22.3	50.9	49.5	12.0	7.6	4.75
LP-YOLO(ours)	PartitionNet	PAN	SIoU	<b>46.8</b>	<b>24.6</b>	52.1	50.0	11.5	6.9	4.52
LP-YOLO(ours)	PartitionNet + VH block + CBAM	PAN	SIoU	<b>48.4</b>	<b>25.1</b>	53.6	49.0	13.9	7.0	4.52
LP-YOLO(ours)	PartitionNet + VH block + CBAM	NewPAN	SIoU	<b>49.0</b>	<b>25.5</b>	<b>53.7</b>	49.8	13.5	7.4	<b>4.32</b>

### 5.3. Performance Comparison Using Different Detection Networks

Table 5 provides a comprehensive comparison of the object detection accuracy between our proposed algorithm and other detection methods. Evidently, our algorithm outperformed the other existing methods with respect to AP50. Furthermore, our analysis revealed that the YOLOv8 model exhibited enhanced performance at higher IoU thresholds.

**Table 5.** Comparing with different detection algorithms.

Model	AP50 (%)	AP50:95 (%)
SSD	38.0	18.5
Faster RCNN	42.5	21.3
YOLOv3	37.7	18.1
YOLOv4	38.1	18.6
YOLOv5-s	46.4	23.0
YOLOv5-m	48.0	25.1
YOLOX-s	43.6	22.8
YOLOX-m	44.6	24.7
YOLOv8-n	46.0	26.3
YOLOv8-s	44.5	26.8
LP-YOLO(ours)	<b>49.0</b>	25.5

#### 5.4. Comparing with the SOTA Lightweight Model GhostNet

Serving as the feature extraction backbone of the detection model also, we made a comparison between the LP-YOLO and the SOTA lightweight model GhostNet. Based on Table 6, the following observations can be made: Our algorithm outperformed GhostNet as backbone of the model in terms of AP50 and AP50:95 with smaller FLOPs on the landslide dataset. We found that adding the VH block and CBAM module to the backbone resulted in a 0.8% AP50 improvement and a 2% AP50:95 improvement for the YOLOv5-GhostNet model. In correspondence with LP-YOLO, it could achieve a 1.6% and a 0.5% improvement for AP50 and AP50:95, respectively. Furthermore, using the NewPAN structure simultaneously could make the model have better performance with a slight increase in FLOPs.

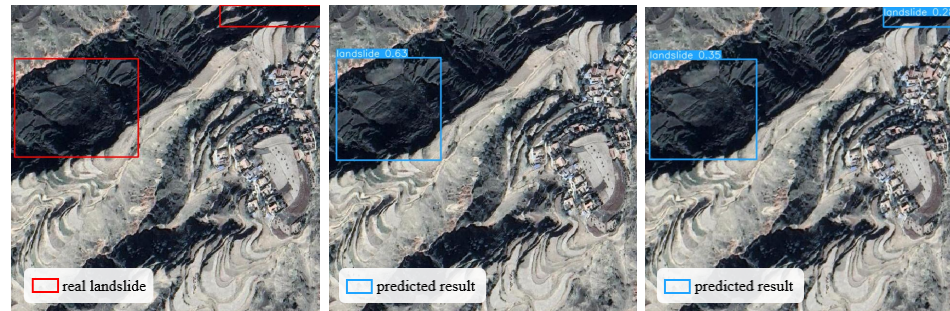
**Table 6.** Comparing with the SOTA lightweight model GhostNet.

Model	Backbone	Neck	AP50 (%)	AP50:95 (%)	FLOPs(G)
YOLOv5	GhostNet	PAN	45.6	22.3	7.6
YOLOv5	GhostNet + VH block + CBAM	PAN	46.4	24.3	7.6
YOLOv5	GhostNet + VH block + CBAM	NewPAN	<b>47.7</b>	<b>25.0</b>	7.9
LP-YOLO	PartitionNet	PAN	46.8	24.6	6.9
LP-YOLO	PartitionNet + VH block + CBAM	PAN	48.4	25.1	7.1
LP-YOLO	PartitionNet + VH block + CBAM	NewPAN	<b>49.0</b>	<b>25.5</b>	7.4

To summarize, in comparison to the original YOLOv5s model, our proposed model in this study exhibited a considerable reduction in the number of parameters and FLOPs, while simultaneously achieving an increase of 2.6% in AP50 and 2.5% in AP50:95. Besides, the proposed model outperformed the mainstream lightweight object-detection algorithms. Specifically, the model attained a detection precision of 49.0% for AP50, 25.5% for AP50:95, and a Latency of 13.5 ms, which fulfilled the real-time detection performance requirements and detection accuracy of landslides, thereby enabling automatic identification of landslides.

Figures 10 and 11 depict the detection performance of YOLOv5s and LP-YOLO on sparse landslide images. Figures 10 and 11 (middle) show the detection results obtained by the YOLOv5s model, while Figures 10 and 11 (right) represent the detection results obtained by the proposed LP-YOLO model. Comparing the two models, it is evident that the YOLOv5s model exhibited a higher confidence score, indicating a greater level of confidence in the detected objects. On the other hand, our proposed LP-YOLO model detected landslides with a slightly lower confidence score, but exhibited a relatively low missed detection rate to strike a balance. Despite the model having a slightly lower

confidence score, the LP-YOLO model displayed higher sensitivity, resulting in fewer instances of missing real objects. This indicated that the model achieved a relatively low false positive rate, even in complex situations, and provided more comprehensive detection results.

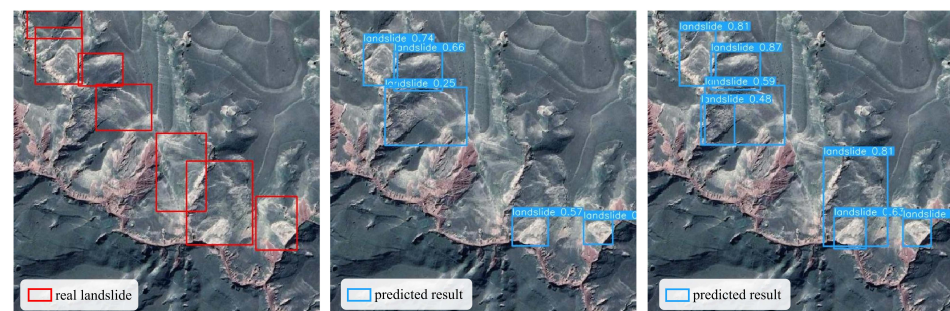


**Figure 10.** Real landslide and predicted results of the YOLOv5s and LP-YOLO models. (Left) Ground truth box for sparse landslide; (middle) YOLOv5s prediction; (right) LP-YOLO model prediction.

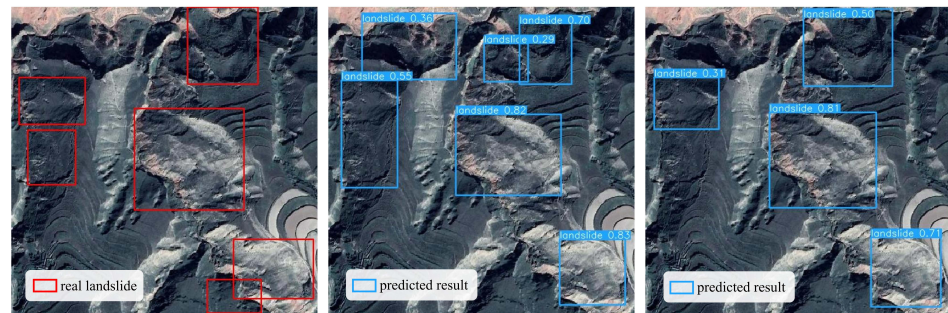


**Figure 11.** Real landslide and predicted results of the YOLOv5s and LP-YOLO models. (Left) Ground truth box for sparse landslide; (middle) YOLOv5s prediction; (right) LP-YOLO model prediction.

Figures 12 and 13 depict the detection performance of YOLOv5s and LP-YOLO on dense landslide images. The detection results indicated that LP-YOLO exhibited superior performance compared to YOLOv5s for dense landslides. However, it is worth noting that the detection performance of LP-YOLO regarding small-area landslides required further enhancement, as there existed a certain missed detection rate.



**Figure 12.** Real landslide and predicted results of the YOLOv5s and LP-YOLO models. (Left) Ground truth box for dense landslide; (middle) YOLOv5s results; (right) LP-YOLO prediction.



**Figure 13.** Real landslide and predicted results of the YOLOv5s and LP-YOLO models. (Left) Ground truth box for dense landslide; (middle) YOLOv5s model prediction; (right) LP-YOLO prediction.

## 6. Discussion

Firstly, the detection results demonstrated that our proposed model exhibited higher precision and average precision (AP) compared to the other models, indicating its ability to achieve a relatively low false positive rate even in complex situations. However, it is worth mentioning that the model detected landslides with relatively low confidence.

Secondly, regarding small target detection, the model designed in this study exhibited limitations in detecting small objects, which is a known challenge in detection tasks. To address this issue, further exploration in future work can involve incorporating multi-scale structures into the model and adopting more-advanced data enhancement methods.

Thirdly, for landslide detection, the scarcity of datasets and the diversity of landslide types contributed to the challenges in achieving accurate identification. For instance, in the case of loess landslides, their colors often closely resemble the surrounding environment, making it difficult for experts to accurately annotate landslide datasets and extract distinctive features for model training.

Lastly, in terms of integration and application in landslide prevention and control systems, the lightweight design of our model offers advantages such as reduced computational burden and memory consumption. These characteristics render it suitable for deployment on power-constrained devices, including unmanned aerial systems (UASs). Integrating the model with UASs would thereby enable efficient and comprehensive monitoring of areas prone to landslides. This integration demonstrates promising prospects in augmenting early warning systems, facilitating real-time monitoring for landslide prevention and management.

## 7. Conclusions

This paper presented LP-YOLO, a new lightweight landslide-detection model based on the YOLO algorithm and a comprehensive landslide dataset. The proposed model addressed the challenges of accurate landslide detection in remote sensing images with varying spectral, texture, terrain, landslide type, and scale characteristics. LP-YOLO comprises several key components that contribute to its superior performance, including the PartitionNet backbone, the VH block, the new feature fusion structure, the CSPCrossStage module, the SIoU loss function, and the CBAM attention mechanism. The experimental results demonstrated that the proposed model outperformed YOLOv5 and the other models in terms of the AP, parameters, and FLOPs. LP-YOLO has the potential to be deployed on power-constrained devices for real-time automatic detection of landslides. However, further research is needed to improve the detection frame rate and model accuracy and expand the dataset to reduce error detection problems. Overall, this work provides a promising solution to the problem of real-time landslide detection and contributes to the development of lightweight detection models for landslides.

**Author Contributions:** Methodology, data curation, software, writing—original draft, validation, P.M., D.L. and M.L.; Formal analysis, writing—review and editing, J.J. and X.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The datasets generated and/or analyzed during the current study are available from the corresponding author on reasonable request.

**Acknowledgments:** We extend our appreciation to Xiaolu Li, Lirong Yin and Wenfeng Zheng for their insightful guidance, constructive criticism, project coordination and support throughout the research project. Their expertise and suggestions have significantly contributed to the quality of this study. We would also like to acknowledge the contributions of our research team members who worked tirelessly to collect and analyze data. Their efforts have been indispensable to the successful completion of this project. We extend our sincere thanks to all those who have contributed to this research project.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Sato, H.P.; Hasegawa, H.; Fujiwara, S.; Tobita, M.; Koarai, M.; Une, H.; Iwahashi, J. Interpretation of landslide distribution triggered by the 2005 Northern Pakistan earthquake using SPOT 5 imagery. *Landslides* **2007**, *4*, 113–122. [\[CrossRef\]](#)
2. Sun, W.; Tian, Y.; Mu, X.; Zhai, J.; Gao, P.; Zhao, G. Loess landslide inventory map based on GF-1 satellite imagery. *Remote Sens.* **2017**, *9*, 314. [\[CrossRef\]](#)
3. Nakano, T.; Wada, K.; Yamanaka, M.; Kamiya, I.; Nakajima, H. Precursory Slope Deformation around Landslide Area Detected by Insar Throughout Japan. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2016**, *41*, 1201. [\[CrossRef\]](#)
4. Huang, C.; Song, K.; Kim, S.; Townshend, J.R.; Davis, P.; Masek, J.G.; Goward, S.N. Use of a dark object concept and support vector machines to automate forest cover change analysis. *Remote Sens. Environ.* **2008**, *112*, 970–985. [\[CrossRef\]](#)
5. Jensen, R.R.; Hardin, P.J.; Yu, G. Artificial neural networks and remote sensing. *Geogr. Compass* **2009**, *3*, 630–646. [\[CrossRef\]](#)
6. Gorsevski, P.V.; Gessler, P.E.; Jankowski, P. A fuzzy k-means classification and a Bayesian approach for spatial prediction of landslide hazard. In *Handbook of Applied Spatial Analysis*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 653–684.
7. Maggiori, E.; Tarabalka, Y.; Charpiat, G.; Alliez, P. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Trans. Geosci. Remote Sens.* **2016**, *55*, 645–657. [\[CrossRef\]](#)
8. Pourghasemi, H.R.; Kerle, N. Random forests and evidential belief function-based landslide susceptibility assessment in Western Mazandaran Province, Iran. *Environ. Earth Sci.* **2016**, *75*, 185. [\[CrossRef\]](#)
9. Tien Bui, D.; Tuan, T.A.; Klempe, H.; Pradhan, B.; Revhaug, I. Spatial prediction models for shallow landslide hazards: A comparative assessment of the efficacy of support vector machines, artificial neural networks, kernel logistic regression, and logistic model tree. *Landslides* **2016**, *13*, 361–378. [\[CrossRef\]](#)
10. Liu, M.; Zhao, F.; Jiang, X.; Zhang, H.; Zhou, H. Parallel binary image cryptosystem via spiking neural networks variants. *Int. J. Neural Syst.* **2022**, *32*, 2150014. [\[CrossRef\]](#)
11. Liu, H.; Liu, M.; Li, D.; Zheng, W.; Yin, L.; Wang, R. Recent advances in pulse-coupled neural networks with applications in image processing. *Electronics* **2022**, *11*, 3264. [\[CrossRef\]](#)
12. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Boston, MA, USA, 7–12 June 2015; pp. 1440–1448.
13. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [\[CrossRef\]](#)
14. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision—ECCV 2016, 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I, No. 14; Springer International Publishing: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
15. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
16. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
17. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
18. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
19. Ju, Y.; Xu, Q.; Jin, S.; Li, W.; Su, Y.; Dong, X.; Guo, Q. Loess landslide detection using object-detection algorithms in northwest China. *Remote Sens.* **2022**, *14*, 1182. [\[CrossRef\]](#)
20. Ji, S.; Yu, D.; Shen, C.; Li, W.; Xu, Q. Landslide detection from an open satellite imagery and digital elevation model dataset using attention boosted convolutional neural networks. *Landslides* **2020**, *17*, 1337–1352. [\[CrossRef\]](#)
21. Hou, H.; Chen, M.; Tie, Y.; Li, W. A Universal Landslide Detection Method in Optical Remote Sensing Images Based on Improved YOLOX. *Remote Sens.* **2022**, *14*, 4939. [\[CrossRef\]](#)



22. Hou, Q.; Zhou, D.; Feng, J. Coordinate attention for efficient mobile network design. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722.
23. Ge, Z.; Liu, S.; Wang, F.; Li, Z.; Sun, J. Yolox: Exceeding yolo series in 2021. *arXiv* **2021**, arXiv:2107.08430.
24. Tang, X.; Tu, Z.; Wang, Y.; Liu, M.; Li, D.; Fan, X. Automatic detection of coseismic landslides using a new transformer method. *Remote Sens.* **2022**, *14*, 2884. [[CrossRef](#)]
25. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Adam, H. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
26. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
27. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
28. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Munich, Germany, 8–14 September 2018; pp. 6848–6856.
29. Han, K.; Wang, Y.; Tian, Q.; Guo, J.; Xu, C.; Xu, C. Ghostnet: More features from cheap operations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 1580–1589.
30. Chen, J.; Kao, S.H.; He, H.; Zhuo, W.; Wen, S.; Lee, C.H.; Chan, S.H.G. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. *arXiv* **2023**, arXiv:2303.03667.
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
32. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
33. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8759–8768.
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [[CrossRef](#)] [[PubMed](#)]
35. Wang, C.Y.; Liao, H.Y.M.; Wu, Y. H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 13–19 June 2020; pp. 390–391.
36. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
37. Elfving, S.; Uchibe, E.; Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Netw.* **2018**, *107*, 3–11. [[CrossRef](#)] [[PubMed](#)]
38. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
39. Liu, Y.; Jia, Q.; Fan, X.; Wang, S.; Ma, S.; Gao, W. Cross-srn: Structure-preserving super-resolution network with cross convolution. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *32*, 4927–4939. [[CrossRef](#)]
40. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 658–666.
41. Gevorgyan, Z. SIoU loss: More powerful learning for bounding box regression. *arXiv* **2022**, arXiv:2205.12740.
42. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. Cbam: Convolutional block attention module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19.
43. Cui, C.; Gao, T.; Wei, S.; Du, Y.; Guo, R.; Dong, S.; Ma, Y. PP-LCNet: A lightweight CPU convolutional neural network. *arXiv* **2021**, arXiv:2109.15099.
44. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning (PMLR), Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
45. Ding, X.; Zhang, X.; Ma, N.; Han, J.; Ding, G.; Sun, J. Repvgg: Making vgg-style convnets great again. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13733–13742.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.